APPENDIX F

Internet Mail Functionality Programming Notes 12/06/98 7:11 AM
doc birth: 12/5/98

Internet mail messaging infrastructure for client software - pseudocode and specs

The Internet mail functionality in the KidCode client software is used to send and receive messages to/from an SMTP/POP3 server on the Internet. Since KidCode is a multiuser client, it includes some mail server-like functionality. If a message recipient is a local user (i.e. a KidCode registered user) then messages are delivered directly to the recipient's mail file.

Sending a message: When a message needs to be sent the KidCode client checks whether the recipient is a local or a remote user. If the recipient is local, the message is written directly to the recipient's mailbox file on the local machine. If the recipient is not a local user, the system opens a SMTP connection and sends the message to the user's SMTP host on the Internet.

Receiving Internet mail: The POP3 protocol is used to receive the user's mail from his Internet POP3 server. When a user goes to open her inbox, the KidCode client opens a connection to the user's POP3 server on the Internet and requests a maildrop. The downloaded messages are MIME decoded and written to the end of the inbox in the user's mail file.

Internet messaging functionality is called in the main email program in the API functions emh_sendMessage and emh_getUserMailbox.

```
--- emh_sendMessage
--- as implemented here, emh_sendMessage opens and closes a
--- SMTP connection to the user's SMTP host each time a message
--- is sent.

on emh_sendMessage kcMessage

if getOne( getRecipient(kcMessage), gmG_RegisteredUsers) then
-- recipient is local
  messageHandler(#sent)

else  -- recipient should be on the Internet
 set retVal = sendSMTPMessage(kcMessage)
end if

if retVal <> 0 then alert("problem sending message to Internet")

end emh_sendMessage
```

```
--- emh_getUserMailbox
--- as implemented here, new mail is read from the server each
--- time the inbox is opened by the user.  Alternatively, email
--- main can be setup to check the POP server periodically for
--- new mail regardless of whether the inbox is opened.

on emh_getUserMailbox mailboxName
global emG_userName, emG_userAddress

set oldBox = readMailbox(mailboxName)

if mailboxName = "inbox" then
     set newMail = getPOPmail()
     append(newMail, oldBox)
end if

return(oldBox)

end emG_getUserMailbox
```

Functions used to implement Internet mail standards for client software:

1. SMTP for sending messages
   a) `sendSMTPMessage`
2. POP3 for receiving maildrops
   a) `getPopMail`
3. minimal MIME compliance for message format
   a) translate KidCode message data structure to MIME complaint string
      i) `makeMimeCompliant`
         a) message header maker/checker
         b) `KCtoMime`
   b) MIME decoder to handle the following encoding schemes and translate the message body back to it's unencoded form
      i) `decodeMime`
         a) base64
         b) quoted-printable
         c) 8 bit
         d) 7 bit

The function sendSMTPMessage is pseudocode to implement the client side of the SMTP protocol. kcMessage is the message to be sent as coded in the kidCode maildata datastructure.

RemoveNextSMTPdata returns the next characters from the mimeMessage to be sent. I assume it returns up to 998 characters each time it is called. If it is called and there are no more characters to be sent, it returns a single ".".

The code structure used below to handleSMTPerror may not work well. I assume that certain errors will cause the transaction to be aborted and the connection closed. This will need to be rethought carefully. I use it here because it makes the code easy to read and understand.

OpenSMTPConnection implements the first part of the SMTP connection protocol...it may be better to incorporate this part of the protocol directly into the sendSMTPMessage function.

Similarly, CloseSMTPConnection implements the last part of the SMTP protocol.

```
on sendSMTPMessage kcMessage
global emG_SMTPServer, emG_userAddress

set mimeMessage = makeMimeCompliant(kcMessage)
set socket = openSMTPConnection(emG_SMTPServer)

-- SMTP accepts SENDER and RECIPIENT fields separately....
-- The remaining MIME message headers are sent as part of the message data.

set retVal = sendLineToSocket("MAIL FROM:" & emG_userAddress
                                          & RETURN & LINEFEED)
if retVal <> "250 OK" then handleSMTPerror(retVal)

--- next only sends a single recipient...it needs to be expanded to
--- handle multiple recipients
set retVal = sendLineToSocket("RCPT TO:" & getRecipient(KCMessage)
                                         & RETURN & LINEFEED)
if retVal <> "250 OK" then handleSMTPerror(retVal)

set retVal = sendLineToSocket("DATA:" & RETURN & LINEFEED)
if retVal <> "250 OK" then handleSMTPerror(retVal)

repeat while nextData <> "." & RETURN & LINEFEED
--- extract the next 998 characters from the messagebody and send them
--- this assumes that removeNextSMTPdata returns a single "."
--- when the message has been completely send, e.g. mimeMessage = ""
      set nextData = removeNextSMTPdata(mimeMessage) & RETURN & LINEFEED
      set retVal = sendLineToSocket(nextdata)
      if retVal <> "250 OK" then handleSMTPerror(retVal)
end repeat

CloseSMTPConnection(socket)

if retVal = "250 OK" then return(1) else return(0)
```

```
end sendSMTPMessage
--- GetPOPMail
--- connects to the user's POP server, gets a maildrop,
--- and returns the messages in a list.

on getPopMail
global emG_POPServer, emG_userAddress

-- opens the POP connection and handles user verification
set socket = openPOPconnection(emG_POPServer, emG_userAddress)

--- get mailDrop returns a list of MIME encoded messages from the
--- POP3 server.
set popMailbox = getmailDrop(socket)
set newMail = []

closePOPconnection()

--- need to decode each message before putting it into the
--- user's local mailbox

repeat with msg in popMailbox
     set kcMessage = decodeMime(msg)
     append(kcMessage, newMail)
end repeat

return(newMail)

end getPopMail
```

```
--- MakeMimeCompliant accepts a Kidcode internal mailData
--- structure and returns a data structure (to be determined)
--- that is a mimeCompliant message including headers and Mime
--- encoded message.  A lot goes on in this function including
--- handling RETURN characters that are part of the message in
--- such a way as to not mess up SMTP...see the MIME specs document
--- for more on this.

on makeMimeComplaint maildata


end makeMimeComplaint


--- decodeMime
--- This function accepts a Mime compliant message and returns a
--- KidCode message data structure to represent the message.
--- The Mime message is decoded (if standard Mime encoding is used).
--- If the content-type of the message is not known, the message
--- body is written to a file in the standard manner of handling
--- attachments. In this event, a text string is written into the
--- message body to indicate the name and location of the file that
--- was written.
--- If the content-type is known, the decoded message body is
--- returned intact in the KidCode message data structure.

on decodeMime mimeMessage


end decodeMime
```